



武汉东湖学院  
Wuhan Donghu College

# 实 验 报 告

学院： 计算机科学学院 专业： 网络空间安全 2025 年 4 月 7 日

姓 名	李畅	学 号	2024040731192
班 级	一班	指导老师	孔德华老师
课程名称	C 语言程序设计实践	成 绩	
实验名称	数组-1		

## 1. 实验目的

1. 理解数组中元素的存储机制；
2. 掌握数据的定义方法；
3. 掌握对数组中元素的赋值和设置方法；

## 2. 实验内容

- 1、将一个长度为 8 的整型数组中的值按逆序存放；（数组中数据自己从键盘动态输入）
- 2、输入 8 个数据，然后按照由小到大的顺序输出；
- 3、从键盘输入一个 4\*3 整型数组赋值，找出其中的最小值，并将该值和其行号与列号输出出来。
- 4、编写一个程序，计算出给定矩阵 a[3][3] 中主对角线元素的和。
- 5、打印出杨辉三角的前 12 行数据，格式为下三角样式；
- 6、输入一个 4\*3 的矩阵，对其转置后输出；
- 7、编写一个程序，把一个数插入到一个有序的有 10 个元素的数组中，并使插入后的数组仍为有序数组。

## 3. 实验环境

- ① windows 10
- ② Vc++6.0/Dev-C++

## 4. 实验方法和步骤（含设计）

1、将一个长度为 8 的整型数组中的值按逆序存放

```
#include<stdio.h>
int main()
{
    int i,a[8];
    printf("请逐个输入元素: \n");
    for(i=7;i>=0;i--)
        scanf("%d",&a[i]);
    for(i=0;i<8;i++)
        printf("%4d",a[i]);
    return 0;
}
```

2、输入 8 个数据，然后按照由小到大的顺序输出：

```
#include <stdio.h>
int main()
{
    int i,j;
    float a[8],Transfer;
    printf("请逐个输入数据并用回车隔开: \n");
    for(i=0;i<8;i++)
    {
        scanf("%f",&a[i]);
    }
    for(i=1;i<=8;i++)
        for(j=0;j<8-i;j++)
            if(a[j]>a[j+1])
            {
                Transfer=a[j] ;
                a[j]=a[j+1] ;
                a[j+1]=Transfer ;
            }
    printf("这八个时候从小到大的排列为 :\n");
    for(i=0;i<8;i++)
        printf("%f\t",a[i]);
    return 0;
}
```

3、从键盘输入一个 4\*3 整型数组赋值，找出其中的最小值，并将该值和其行号与列号输出出来。

```
#include<stdio.h>
int main()
{
    int i,j,k=1,min,min1,min2,a[4][3];
    printf("请按以下格式输入四次数组数据: \n");
    printf("格式为:  %d,%d,%d \n");
    for(i=0;i<4;i++)
        scanf("%d,%d,%d",&a[i][0],&a[i][1],&a[i][2]);
    min=a[0][0];min1=0;min2=0;
    for(i=0;i<4;i++)
        for(j=0;j<3;j++)
            if(min>a[i][j])
                {min=a[i][j];min1=i;min2=j;}
    printf("该矩阵中的最小值为: %d, 它在第 %d 行, 第 %d 列",min,min1+1,min2+1);
    return 0;
}
```

4、计算出给定矩阵 a[3][3] 中主对角线元素的和。

```
#include<stdio.h>
int main()
{
    int i;
    float sum=0,a[3][3];
    printf("请按以下格式输入三次数据: \n");
    printf("格式为:  %f,%f,%f  \n");
    for(i=0;i<3;i++)
        scanf("%f,%f,%f",&a[i][0],&a[i][1],&a[i][2]);
    for(i=0;i<3;i++)
        sum+=a[i][i];
    printf("该矩阵主对角元素之和为: %f",sum);
    return 0;
}
```

5、打印出杨辉三角的前 12 行数据，格式为下三角样式；

```
#include<stdio.h>
int main()
{
    int i,j,a[12][12];
    for(i=0;i<12;i++)
        a[i][0]=a[i][i]=1;
    for(i=0;i<11;i++)
        for(j=1;j<=i;j++)
            a[i+1][j]=a[i][j-1]+a[i][j];
    for(i=0;i<12;i++)
    {
        for(j=0;j<=i;j++)
            printf("%d\t",a[i][j]);
        printf("\n");
    }
    return 0;
}
```

6、输入一个 4\*3 的矩阵，对其转置后输出；

```
#include<stdio.h>
int main()
{
    int i,j,k=1,a[4][3],b[3][4];
    printf("请按以下格式输入四次数据: \n");
    printf("格式为:  %d,%d,%d  \n");
    for(i=0;i<4;i++)
        scanf("%d,%d,%d",&a[i][0],&a[i][1],&a[i][2]);
    for(i=0;i<4;i++)
        for(j=0;j<3;j++)
            b[j][i]=a[i][j];
    printf("该矩阵的转置为: \n");
    for(i=0;i<3;i++)
        for(j=0;j<4;j++)
        {
            printf("%d,",b[i][j]);
            if(k++%4==0)
                printf("\n");
        }
    return 0;
}
```

7、编写一个程序，把一个数插入到一个有序的有 10 个元素的数组中，并使插入后的数组仍为有序数组。

```
1  #include<stdio.h>
2  int main()
3  {
4      int i,t,n,m=0,key=0,a[11];
5      printf("请逐个输入十个有序元素: \n");
6      for(i=0;i<10;i++)
7          scanf("%d",&a[i]);
8      for(;key==0;)
9      {
10         key=1;
11         for(i=0;i<9;i++)
12             key*=(a[i]<=a[i+1]);
13         for(i=0;i<9;i++)
14             if(a[i]>a[i+1])
15                 {t=a[i];a[i]=a[i+1];a[i+1]=t;m++;}
16     }
17     if(m)
18         printf("你个坏人不老实!!! \n");
19     printf("十个元素构成的有序数组为: \n");
20     for(i=0;i<10;i++)
21         printf("%d\t",a[i]);
22     printf("\n请输入待插入的元素: \n");
23     scanf("%d",&n);
24     for(i=9;i>=0;i--)
25     {
26         if(n<a[i])
27             a[i+1]=a[i];
28         else
29         {
30             a[i+1]=n;
31             break;
32         }
33     }
34     printf("十一个元素构成的有序数组为: \n");
35     for(i=0;i<11;i++)
36         printf("%d\t",a[i]);
37     return 0;
38 }
```

## 5. 程序及测试结果

1、将一个长度为 8 的整型数组中的值按逆序存放

请逐个输入元素：

```
1
2
3
5
6
8
9
7
  7  9  8  6  5  3  2  1
```

-----  
Process exited after 20.08 seconds with return value 0  
请按任意键继续. . .

2、输入 8 个数据，然后按照由小到大的顺序输出；

请逐个输入数据并用回车隔开：

```
1
2
5
7
8
3
6
9
这八个时候从小到大的排列为：
1.000000    2.000000    3.000000    5.000000    6.000000
7.000000    8.000000    9.000000
```

-----  
Process exited after 5.253 seconds with return value 0  
请按任意键继续. . .

3、从键盘输入一个 4\*3 整型数组赋值，找出其中的最小值，并将该值和其行号与列号输出出来。

请按以下格式输入四次数据：

格式为： %d,%d,%d

```
1,2,3
5,7,9
0,2,9
9,-1,4
该矩阵中的最小值为：-1，它在第 4 行，第 2 列
```

-----  
Process exited after 21.97 seconds with return value 0  
请按任意键继续. . .

4、计算出给定矩阵 a[3][3] 中主对角线元素的和。

请按以下格式输入三次数据：

格式为： %f,%f,%f

```
1,5,6
7,5,4
22,4,6
该矩阵主对角元素之和为：12.000000
```

-----  
Process exited after 13.97 seconds with return value 0  
请按任意键继续. . .

5、打印出杨辉三角的前 12 行数据，格式为下三角样式；

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
1 8 28 56 70 56 28 8 1
1 9 36 84 126 126 84 36 9 1
1 10 45 120 210 252 210 120 45 10 1
1 11 55 165 330 462 462 330 165 55 11 1
```

-----  
Process exited after 0.617 seconds with return value 0  
请按任意键继续. . .

6、输入一个 4\*3 的矩阵，对其转置后输出；

请按以下格式输入四次数据：  
格式为： %d,%d,%d

1,2,3  
55,4,6  
-1,22,4  
9,8,7

该矩阵的转置为：

1,55,-1,9,  
2,4,22,8,  
3,6,4,7,

-----  
Process exited after 20.08 seconds with return value 0  
请按任意键继续. . .

7、编写一个程序，把一个数插入到一个有序的有 10 个元素的数组中，并使插入后的数组仍为有序数组。

请逐个输入十个有序元素：

```
1
2
3
4
5
6
7
8
0
9
```

你个坏人不老实!!!

十个元素构成的有序数组为：

0 1 2 3 4 5 6 7 8 9

请输入待插入的元素：

5

十一个元素构成的有序数组为：

0 1 2 3 4 5 5 6 7 8 9

-----  
Process exited after 26.63 seconds with return value 0  
请按任意键继续. . .

## 6. 实验分析与体会

通过本次实验，我深入理解了数组在程序设计中的核心作用。从一维数组的逆序存储到二维矩阵的转置操作，每一个任务都让我对数组的存储机制和操作逻辑有了更直观的认识。在实现数组逆序时，我通过倒序输入直接完成逆序存储，虽结果正确，但也意识到传统“首尾交换”方法更能体现算法思维，这让我反思代码逻辑与实际需求的匹配性。动态输入数据的过程中，我掌握了如何通过循环结构灵活控制数组赋值，并深刻体会到输入格式提示的重要性——例如在  $4 \times 3$  矩阵输入时明确要求“a, b, c”格式，避免因用户操作错误导致数据错位。

对数组排序的实践让我真正理解了冒泡排序的底层逻辑。通过相邻元素的反复比较与交换，我直观感受到算法的时间复杂度对性能的影响，尤其是在处理大规模数据时的效率瓶颈。而在二维数组中寻找最小值时，双重循环的嵌套遍历让我熟悉了行列索引的协同控制，同时也意识到初始化条件的重要性：初始若将最小值设为第一个元素，后续比较中需严格使用“小于”而非“小于等于”，否则可能记录到最后一个重复值而非真正的最小值。

矩阵操作是本次实验的另一个重点。主对角线求和的实现让我掌握了如何利用“ $i=j$ ”的索引规律快速定位元素，而矩阵转置则通过行列索引互换揭示了数据重组的本质。这些操作让我体会到数组在数学计算中的实用性，例如转置在线性代数中的应用。此外，在有序数组插入元素的实验中，我从后向前遍历并逐个后移元素的操作，既巩固了对数组连续存储特性的理解，也让我意识到固定长度数组的局限性——若插入次数频繁，需结合动态内存管理提升灵活性。

调试代码的过程同样让我受益匪浅。例如，浮点数输出未控制小数位导致显示混乱，让我学会用“%.2f”规范格式；冒泡排序外层循环次数设置错误引发越界风险，则加深了我对循环边界的敏感性。这些细节问题让我认识到编程不仅是逻辑构建，更是对每一行代码的细致打磨。通过本次实验，我不仅掌握了数组的基础操作，更在实践中培养了严谨的编程思维，为后续学习更复杂的数据结构奠定了坚实基础。

实验日期： 2025 年 4 月 7 日

教师评语

签名： 年 月 日